File Streams

Vulnerable to TOCTOU issues

Sean Barnum, Cigital, Inc. [vita[1]]

2007-03-22

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 6375 bytes

Attack Category	• Path spoofing or confusion problem
Vulnerability Category	• Indeterminate File/Path • TOCTOU - Time of Check, Time of Use
SoftwareContext	• File I/O
Location	• studio.h • stropts.h
Description	Care must be exercised when accessing files from passed in pathnames. Each of these functions takes a filename (or pathname) that can potentially be subjected to a time-of-check-to-time-of-use race condition during an access control check. fattach(int fildes, const char *path) - associates a pathname with a file descriptor, filedes, by attaching a STREAMS-based file descriptor to the file. fdetach(const char *path) - detaches a STREAMS-based file from the file pointed to by path. fopen(const char *filename, const char *mode) - associates a stream with the file pointed to by filename. These functions are all vulnerable to TOCTOU issues where the filename/pathname could be changed, linked or spoofed when the actual call is made.
APIs	**FunctionName** / **Comments** fattach fdetach fopen
Method of Attack	An attacker can leverage these API to attach/detach a stream with an arbitrary file and possibly obtain secret information or elevate their privileges. The key issue with respect to TOCTOU vulnerabilities is that programs make assumptions

1. http://buildsecurityin.us-cert.gov/bsi-rules/35-BSI.html (Barnum, Sean)

	about atomicity of actions. It is assumed that checking the state or identity of a targeted resource followed by an action on that resource is all one action. In reality, there is a period of time between the check and the use that allows either an attacker to intentionally or another interleaved process or thread to unintentionally change the state of the targeted resource and yield unexpected and undesired results An attacker could link the targeted path to a known path of the attackers choosing allowing the attacker access to files that he should not have access to.
Exception Criteria	
Solutions	

Solution Applicability	Solution Description	Solution Efficacy
Generally applies.	The most basic advice for TOCTOU vulnerabilities is to not perform a check before the use. This does not resolve the underlying issue of the execution of a function on a resource whose state and identity cannot be assured, but it does help to limit the false sense of security given by the check.	Does not resolve the underlying vulnerability but limits the false sense of security given by the check.
Generally applies.	Limit the interleaving of operations on files from multiple processes.	Does not eliminate the underlying vulnerability but can help make it more difficult to exploit.
Generally applies.	Limit the spread of time (cycles) between the check and use of a resource.	Does not eliminate the underlying vulnerability but can help make it more

			difficult to exploit.
	Generally applies.	Recheck the resource after the use call to verify that the action was taken appropriately.	Checking the filename permissions after the operation does not change the fact that the operation may have been exploited but it does allow halting of the application in an error state to help limit further damage.
	When user specification of the file to be altered is not necessary.	Do not rely on user-specified input to determine what the file to be made executable.	This will reduce exposure but will not eliminate the problem.

Signature Details

Examples of Incorrect Code

```
#include
#include

/* check permissions to the file*/

int filedes;
fileDes = getFileDes(...);

if(!access(file, ...){
/* attach a STREAMS-based file
descriptor to the file */
fattach(fileDes, pathname);
}
else{
/* permission was denied */
}
```

Examples of Corrected Code

Source References

- http://www.opengroup.org/onlinepubs/007908799/xsh/fattach.html
- http://www.opengroup.org/onlinepubs/007908799/xsh/fdetach.html
- http://www.opengroup.org/onlinepubs/007908799/xsh/fopen.html
- ITS4 Source Code Vulnerability Scanning Tool [5]

	• Viega, John & McGraw, Gary. Building Secure Software: *How to Avoid Security Problems the Right Way*. Boston, MA: Addison-Wesley Professional, 2001, ISBN: 020172152X
Recommended Resource	
Discriminant Set	**Operating System**: • ~~AIX as~~ **~~placeholder~~** **Language**

Cigital, Inc. Copyright

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. mailto:copyright@cigital.com